

|  |
| --- |
| « 2048 » |

*Une image contenant table, calendrier

Description générée automatiquement*

Ozdemir Fakime

Fakime-nur.ozdemir@cpnv.ch

Table des matières

SI-C1b

24.03.2023

[1 Introduction 3](#_Toc125451210)

[1.1 Cadre, description et motivation 3](#_Toc125451211)

[1.2 Objectifs 3](#_Toc125451212)

[1.3 Planification initiale 3](#_Toc125451213)

[2 Analyse 3](#_Toc125451214)

[2.1 Maquettes 3](#_Toc125451215)

[2.2 Stories / tests d’acceptation 3](#_Toc125451216)

[2.3 Stratégie de test 3](#_Toc125451217)

[3 Implémentation 3](#_Toc125451218)

[3.1 Vue d’ensemble 3](#_Toc125451219)

[3.2 Choix techniques 3](#_Toc125451220)

[3.3 Points techniques spécifiques 3](#_Toc125451221)

[3.3.1 Point 1 3](#_Toc125451222)

[3.3.2 Point 2 3](#_Toc125451223)

[3.3.3 Point … 3](#_Toc125451224)

[4 Tests 3](#_Toc125451225)

[4.1 Tests effectués 3](#_Toc125451226)

[4.2 Erreurs restantes 3](#_Toc125451227)

[5 Conclusions 3](#_Toc125451228)

[6 Annexes 3](#_Toc125451229)

[6.1 Sources – Bibliographie 3](#_Toc125451230)

[6.2 Journal de bord du projet 3](#_Toc125451231)

# Introduction

## Cadre, description et motivation

Dans le cadre du CPNV, dans le module ICT-431, enseigné par Jean Philippe Chavey, je réalise la programmation du jeu 2048.

## Objectifs

Sprint 1 : Une maquette du jeu, un code avec deux exemples de tableaux/listes en mémoire. Des tuiles de toutes les valeurs de 2 à 8192 (=13 cases, de 21 é 213) avec leur couleur.

Sprint 2 : Un tableau de 4 valeurs « tassées » vers le début du tableau.

Mon objectif est de terminer toutes ces étapes et de livrer ce programme avec le moins d'erreurs possible. Et en même temps, voir mes erreurs et faire plus attention.

A la fin je veux avoir un jeu fluide et agréable à jouer pour toutes personnes, de bien réussir le jeu avec tout c’est mouvement.

## Planification initiale

Les étapes sont séparées en sprint. Les sprints ont chaqu’un plusieurs étapes.

Sprint 1 : du 10/02/2023

Sprint 2 : 10/03/2023

Sprint 3 : 24/05/2023

Défense 30-31 mars et 6 avril, démonstration devant la classe.

# Analyse

## Maquettes

Une image contenant graphique

Description générée automatiquement

## Stories / tests d’acceptation

Les tests ont été réalisés sur IceScrum :

https://icescrum.cpnv.ch/p/MA202320O/#/planning/3124/sprint/3126/details

## Stratégie de test

Option de test sur iceScrum et moodle.

# Implémentation

## Vue d’ensemble

Ce programme fonctionne sur un ordinateur et interagit avec les touches de clavier « a », « s »,« d », « w » pour les déplacements.

L’interface a été faite avec python et tkinter.

## Choix techniques

Je suis sur un système d’exploitation MacOs, j’ai donc du installer une virtuel box (Parallels Desktop) et pour programmer j’utilise PyCharm 2022.

J’utilise aussi github pour stocker et partager le code et icescrum pour la planification du projet.

## Points techniques spécifiques

Je veux vous expliquer la fonctionne qui sert a tasser nos chiffres :

Cette fonction était une fonction difficile pour moi car j'avais du mal à comprendre sa logique. Mais après avoir compris la logique, j'ai réalisé que ce n’était pas si compliqué.

\*\*Pour explique mieux j'ai été aidé par un informaticien turc.

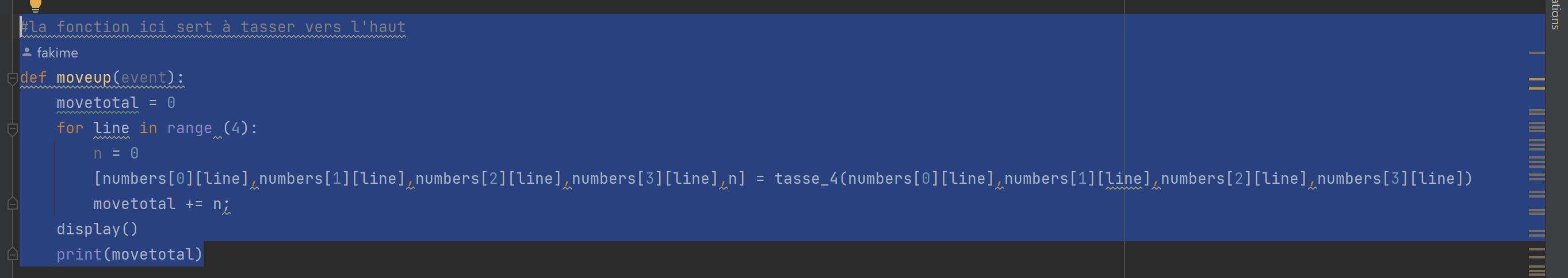
Cette fonction sert à déplacer les chiffres dans une "table" de 4 colonnes et 4 lignes vers le haut. Elle commence par initialiser une variable movetotal à 0, qui servira à compter le nombre de mouvements nécessaires pour déplacer les chiffres. Ensuite, elle parcourt chaque ligne de la table à l'aide de la boucle for.

Pour chaque ligne, la fonction appelle une autre fonction appelée tasse\_4 avec les quatre chiffres de la ligne en question. La fonction tasse\_4 sert à déplacer les chiffres vers le haut autant que possible tout en fusionnant les chiffres identiques. La fonction tasse\_4 renvoie la nouvelle ligne de chiffres ainsi que le nombre de mouvements effectués pour déplacer les chiffres.

La fonction moveup récupère la nouvelle ligne de chiffres retournée par tasse\_4 et l'assigne à l'emplacement approprié dans la table. Elle utilise également la variable n retournée par tasse\_4 pour ajouter au total de mouvements movetotal.

Enfin, la fonction moveup appelle une autre fonction appelée display pour afficher la table mise à jour et imprime le nombre total de mouvements effectués pour déplacer les chiffres.

En résumé, la fonction moveup sert à déplacer les chiffres d'une table de 4 colonnes et 4 lignes vers le haut tout en fusionnant les chiffres identiques, et elle affiche le nombre total de mouvements effectués pour y parvenir.



Ensuite je veux vous expliquer la fonctionne de tasse4 :

Cette fonction manipule ces nombres en les déplaçant vers la gauche et en fusionnant des nombres identiques. Elle renvoie ensuite une liste de 5 éléments contenant les nouveaux nombres ainsi que le nombre de déplacements nécessaires pour y parvenir.

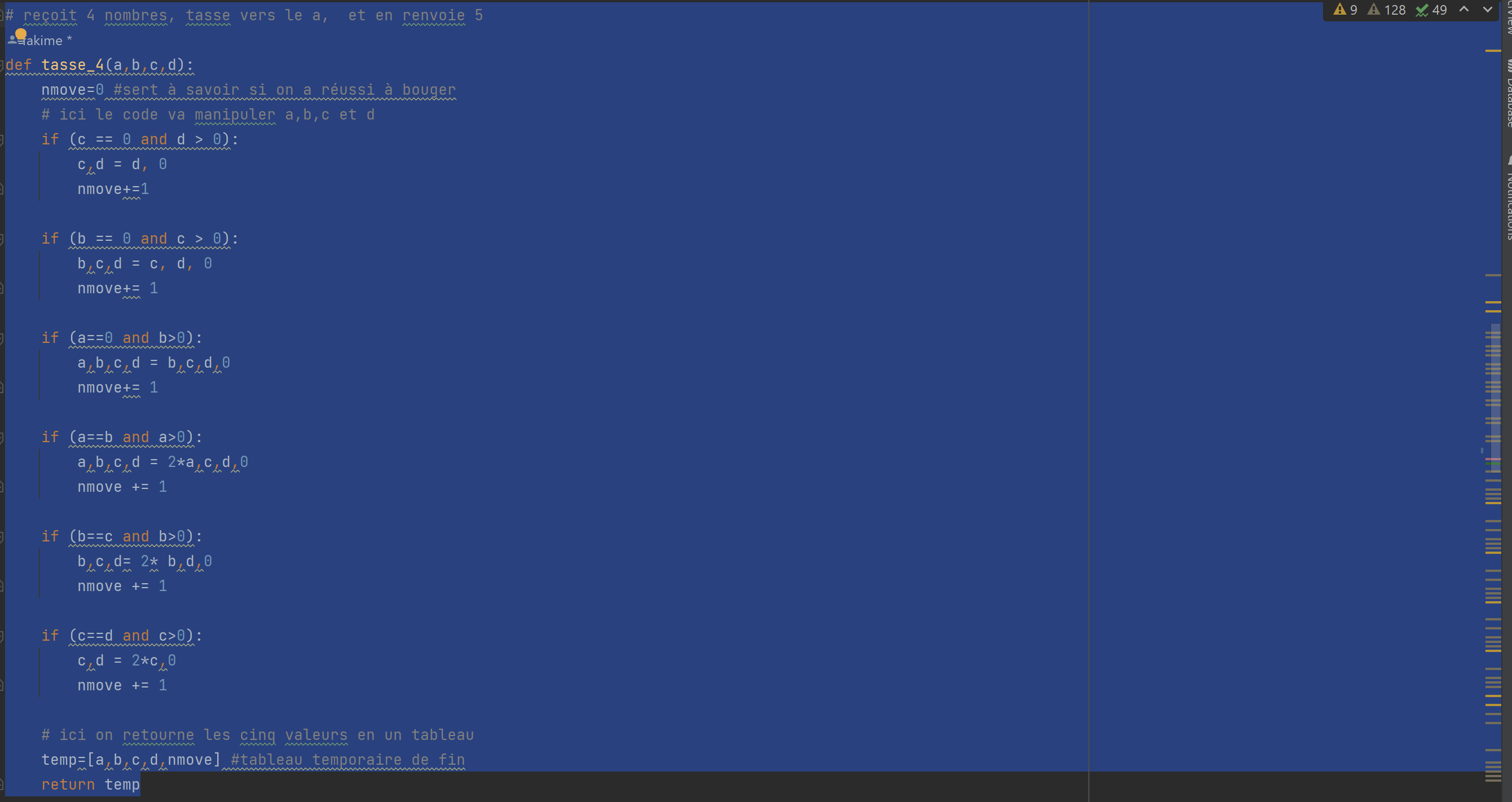
Le code commence par initialiser une variable nmove à 0, qui sera utilisée pour compter le nombre de mouvements nécessaires pour déplacer les nombres. Ensuite, il y a plusieurs blocs if qui déplacent les nombres en fonction de leur position actuelle dans l'ordre a, b, c et d.

Si le nombre c est 0 et d est plus grand que 0, le code déplace d dans la position de c et met 0 dans la position de d. De même, si b est 0 et c est plus grand que 0, le code déplace c dans la position de b, d dans la position de c, et met 0 dans la position de d.

Le code poursuit cette logique pour les nombres a et b, a et b, b et c, et c et d. Si deux nombres adjacents sont identiques, le code les fusionne en doublant le nombre et en mettant 0 dans la position de l'autre nombre.

Finalement, la fonction tasse\_4 renvoie les quatre nombres dans un tableau ainsi que la variable nmove qui contient le nombre total de mouvements effectués pour déplacer les nombres.

Et enfin, cette fonction sert à déplacer quatre nombres vers la gauche et à fusionner des nombres identiques, en renvoyant une liste de 5 éléments contenant les nouveaux nombres et le nombre total de mouvements effectués pour y parvenir.



Pour le sprint 3 je vais vous expliquer La fonction **spawncase2\_4()** que vous avez fournie génère un nombre aléatoire soit 2, soit 4, avec une chance de 80% d'obtenir un 4 et de 20% d'obtenir un 2. La probabilité de générer un 4 est donc plus élevée que celle de générer un 2.

Si vous souhaitez modifier les probabilités de manière à ce que la fonction génère un 2 avec une probabilité de 80% et un 4 avec une probabilité de 20%, vous pouvez inverser les seuils dans la condition **if**.



### Point 1

### Point 2

### Point …

# Tests

## Tests effectués

https://icescrum.cpnv.ch/p/MA202320O/#/planning/3124/sprint/3126/details

## Erreurs restantes

Tous les erreurs réparés, programme sans erreur lors de rendre le sprint 2.

# Conclusions

Ces codes implémentent le jeu 2048 dans une fenêtre graphique à l'aide de la bibliothèque tkinter de Python. La grille de jeu est affichée à l'aide de 16 étiquettes qui sont mises à jour à chaque tour de jeu. Les mouvements sont gérés par des fonctions qui déplacent les cases de la grille dans la direction souhaitée. Le jeu est gagné lorsque l'une des cases atteint la valeur de 2048, et perdu lorsque toutes les cases sont remplies et qu'aucun mouvement n'est possible.

La fonction principale tasse\_4 déplace les cases de la grille vers la gauche, la droite, le haut ou le bas en fonction de la direction souhaitée. Elle combine également les cases identiques et met à jour le score en conséquence. La fonction movement appelle la fonction tasse\_4 pour déplacer toutes les cases de la grille dans la direction souhaitée. Elle vérifie également si le joueur a perdu en ne pouvant plus effectuer de mouvements.

Le jeu est contrôlé par des événements de clavier qui appellent les fonctions de mouvement appropriées. Le score est affiché dans une étiquette en temps réel et une boîte de dialogue s'affiche lorsque le joueur gagne ou perd la partie.

En fin de compte, ces codes implémentent avec succès le jeu 2048 et offrent une expérience de jeu interactive pour l'utilisateur.

# Annexes

## Sources – Bibliographie

Youtube : <https://www.youtube.com/watch?v=7EIvMsg5AWU&list=PL-Hkw4CrSVq9Y_RP7Q9Kn-bgZvVdl1cBy>

Courses : https://www.freecodecamp.org/learn/scientific-computing-with-python/

Sites Web utiles : https://python.doctor/page-apprendre-creer-fonction-en-python

## Journal de bord du projet

|  |  |
| --- | --- |
| **Date** | **Evénement** |
|  |  |

## Journal de travail

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Jour** | **Semaine** | **Temps [h]** | **Type** | **Description** | **Remarques** |
| 01.02.2023 | 1 | 1,5h |  | Maquette |  |
| 03.02.2023 | 1 | 5h |  | Affichage |  |
| 08.02.2023 | 2 | 1,5 |  | Affichage |  |
| 10.02.2023 | 2 | 5h |  | Rendre maquette + projet avec le fonctions |  |
| 22.02.2023 | 3 | 1h |  | Avancement dans le programme |  |
| 24.02.2023 | 3 | 4h |  | Avancement dans le programme + iceScrum |  |
| 01.03.2023 | 4 | 1,5h |  | Avancement dans le programme + iceScrum + Github |  |
| 03.03.2023 | 4 | - |  | Absente |  |
| 07.03.2023 | 5 | - |  | Absente |  |
| 10.03.2023 | 5 | 5h |  | Fin sprint + Rendre |  |
| 15.03.2023 | 6 | 1,5h |  | Avancement dans le programme |  |
| 17.03.2023 | 6 | 5h |  | Avancement dans le programme + iceScrum + Github |  |
| 22.03.2023 | 7 | 1,5h |  | Avancement dans le programme + iceScrum |  |
| 24.03.2023 | 7 | 5h |  | Fin de sprint + rendre le prog |  |